

BBBBBBBBBBBB		AAAAAAA		SSSSSSSSSS		RRRRRRRRRR		TTTTTTTTTTTT		LLL
BBBBBBBBBBBB		AAAAAAA		SSSSSSSSSS		RRRRRRRRRR		TTTTTTTTTTTT		LLL
BBBBBBBBBBBB		AAAAAAA		SSSSSSSSSS		RRRRRRRRRR		TTTTTTTTTTTT		LLL
BBB	BBB	AAA	AAA	SSS		RRR	RRR	TTT		LLL
BBB	BBB	AAA	AAA	SSS		RRR	RRR	TTT		LLL
BBB	BBB	AAA	AAA	SSS		RRR	RRR	TTT		LLL
BBB	BBB	AAA	AAA	SSS		RRR	RRR	TTT		LLL
BBB	BBB	AAA	AAA	SSS		RRR	RRR	TTT		LLL
BBB	BBB	AAA	AAA	SSS		RRR	RRR	TTT		LLL
BBBBBBBBBBBB		AAA	AAA	SSSSSSSS		RRRRRRRRRR		TTT		LLL
BBBBBBBBBBBB		AAA	AAA	SSSSSSSS		RRRRRRRRRR		TTT		LLL
BBBBBBBBBBBB		AAA	AAA	SSSSSSSS		RRRRRRRRRR		TTT		LLL
BBB	BBB	AAAAAAAAAAAA			SSS	RRR	RRR	TTT		LLL
BBB	BBB	AAAAAAAAAAAA			SSS	RRR	RRR	TTT		LLL
BBB	BBB	AAAAAAAAAAAA			SSS	RRR	RRR	TTT		LLL
BBB	BBB	AAA	AAA		SSS	RRR	RRR	TTT		LLL
BBB	BBB	AAA	AAA		SSS	RRR	RRR	TTT		LLL
BBB	BBB	AAA	AAA		SSS	RRR	RRR	TTT		LLL
BBB	BBB	AAA	AAA		SSS	RRR	RRR	TTT		LLL
BBB	BBB	AAA	AAA		SSS	RRR	RRR	TTT		LLL
BBBBBBBBBBBB		AAA	AAA	SSSSSSSSSS		RRR	RRR	TTT		LLLLLLLLLLLL
BBBBBBBBBBBB		AAA	AAA	SSSSSSSSSS		RRR	RRR	TTT		LLLLLLLLLLLL
BBBBBBBBBBBB		AAA	AAA	SSSSSSSSSS		RRR	RRR	TTT		LLLLLLLLLLLL

```
BBBBBBBBB      AAAAAA      SSSSSSSSS      000000      PPPPPPPP      EEEEEEEEEEE      NN      NN      ZZZZZZZZZZ      EEEEEEEEEEE
BBBBBBBBB      AAAAAA      SSSSSSSSS      000000      PPPPPPPP      EEEEEEEEEEE      NN      NN      ZZZZZZZZZZ      EEEEEEEEEEE
BB      BB      AA      AA      SS      SS      00      00      PP      PP      EE      NN      NN      ZZ      EE
BB      BB      AA      AA      SS      SS      00      00      PP      PP      EE      NN      NN      ZZ      EE
BB      BB      AA      AA      SS      SS      00      00      PP      PP      EE      NNNN      NN      ZZ      EE
BBBBBBBBB      AA      AA      SSSSSSS      00      00      PPPPPPPP      EEEEEEEEE      NN      NN      ZZ      EEEEEEEEE
BBBBBBBBB      AA      AA      SSSSSSS      00      00      PPPPPPPP      EEEEEEEEE      NN      NN      ZZ      EEEEEEEEE
BB      BB      AAAAAAAAAA      SS      00      00      PP      EE      NN      NNNN      ZZ      EE
BB      BB      AAAAAAAAAA      SS      00      00      PP      EE      NN      NNNN      ZZ      EE
BB      BB      AA      AA      SS      00      00      PP      EE      NN      NN      ZZ      EE
BB      BB      AA      AA      SS      00      00      PP      EE      NN      NN      ZZ      EE
BBBBBBBBB      AA      AA      SSSSSSS      000000      PP      EEEEEEEEEEE      NN      NN      ZZZZZZZZZZ      EEEEEEEEEEE
BBBBBBBBB      AA      AA      SSSSSSS      000000      PP      EEEEEEEEEEE      NN      NN      ZZZZZZZZZZ      EEEEEEEEEEE
```

```
LL      IIIIII      SSSSSSSSS
LL      IIIIII      SSSSSSSSS
LL      II      SS
LL      II      SS
LL      II      SS
LL      II      SS
LL      II      SSSSS
LL      II      SSSSS
LL      II      SS
LL      II      SS
LL      II      SS
LLLLLLLLLLLL      IIIIII      SSSSSSSSS
LLLLLLLLLLLL      IIIIII      SSSSSSSSS
```



```
1 0001 0 MODULE BAS$$OPEN_ZERO (
2 0002 0 IDENT = '1-002'
3 0003 0 ) =
4 0004 1 BEGIN
5 0005 1
6 0006 1 *****
7 0007 1 *
8 0008 1 * COPYRIGHT (c) 1978, 1980, 1982, 1984 BY
9 0009 1 * DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.
10 0010 1 * ALL RIGHTS RESERVED.
11 0011 1 *
12 0012 1 * THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED
13 0013 1 * ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE
14 0014 1 * INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER
15 0015 1 * COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY
16 0016 1 * OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY
17 0017 1 * TRANSFERRED.
18 0018 1 *
19 0019 1 * THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
20 0020 1 * AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT
21 0021 1 * CORPORATION.
22 0022 1 *
23 0023 1 * DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS
24 0024 1 * SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.
25 0025 1 *
26 0026 1 *
27 0027 1 *****
28 0028 1
29 0029 1
30 0030 1 ++
31 0031 1 FACILITY: BASIC-PLUS-2 Miscellaneous I/O
32 0032 1
33 0033 1 ABSTRACT:
34 0034 1
35 0035 1 This module contains an internal subroutine used by several
36 0036 1 of the BASIC functions which can operate on the terminal.
37 0037 1 The BASIC language definition assumes that the terminal is
38 0038 1 "always open", but on VAX we do not open it until we need to.
39 0039 1 To make this as easy as possible, this module OPENS channel
40 0040 1 zero whenever anyone needs it.
41 0041 1
42 0042 1 ENVIRONMENT: VAX-11 User Mode
43 0043 1
44 0044 1 AUTHOR: John Sauter, CREATION DATE: 17-APR-1979
45 0045 1
46 0046 1 MODIFIED BY:
47 0047 1
48 0048 1 1-001 - Original.
49 0049 1 1-002 - Set ISB$A_USER_FP. JBS 25-JUL-1979
50 0050 1 --
51 0051 1
52 0052 1 !<BLF/PAGE>
```



```
54 0053 1 |
55 0054 1 | SWITCHES:
56 0055 1 |
57 0056 1 |
58 0057 1 | SWITCHES ADDRESSING_MODE (EXTERNAL = GENERAL, NONEXTERNAL = WORD_RELATIVE);
59 0058 1 |
60 0059 1 |
61 0060 1 | LINKAGES:
62 0061 1 |
63 0062 1 |
64 0063 1 | REQUIRE 'RTLIN:OTSLNK'; | Define linkages
65 0492 1 |
66 0493 1 |
67 0494 1 | TABLE OF CONTENTS:
68 0495 1 |
69 0496 1 |
70 0497 1 | FORWARD ROUTINE
71 0498 1 |     BASS$OPEN_ZERO : NOVALUE; | Open channel zero
72 0499 1 |
73 0500 1 |
74 0501 1 | INCLUDE FILES:
75 0502 1 |
76 0503 1 |
77 0504 1 | REQUIRE 'RTLML:OTSLUB'; | Get LUB definitions
78 0644 1 |
79 0645 1 | REQUIRE 'RTLML:OTISISB'; | Get ISB definitions
80 0813 1 |
81 0814 1 | REQUIRE 'RTLIN:RTLPSECT'; | Macros for defining psects
82 0909 1 |
83 0910 1 | LIBRARY 'RTLSTARLE'; | System symbols
84 0911 1 |
85 0912 1 |
86 0913 1 | MACROS:
87 0914 1 |
88 0915 1 |     NONE
89 0916 1 |
90 0917 1 | EQUATED SYMBOLS:
91 0918 1 |
92 0919 1 |     NONE
93 0920 1 |
94 0921 1 | PSECTS:
95 0922 1 |
96 0923 1 | DECLARE_PSECTS (BAS); | Declare psects for BASS$ facility
97 0924 1 |
98 0925 1 | OWN STORAGE:
99 0926 1 |
100 0927 1 |     NONE
101 0928 1 |
102 0929 1 | EXTERNAL REFERENCES:
103 0930 1 |
104 0931 1 |
105 0932 1 | EXTERNAL ROUTINE
106 0933 1 |     BASS$CB_PUSH : JSB_CB_PUSH NOVALUE, | Load register CCB
107 0934 1 |     BASS$CB_POP : JSB_CB_POP NOVALUE, | Done with register CCB
108 0935 1 |     BASS$OPEN_DEFLT : CALL_CCB NOVALUE; | Open one side of chan. 0
109 0936 1 |
```

```

111 0937 1 GLOBAL ROUTINE BASS$OPEN_ZERO (
112 0938 1     FMP
113 0939 1     ) : NOVALUE =
114 0940 1
115 0941 1  !+
116 0942 1  ! Open channel zero
117 0943 1  ! User's frame pointer
118 0944 1  FUNCTIONAL DESCRIPTION:
119 0945 1  Opens BASIC 'channel 0', which is implemented as two LUNs,
120 0946 1  linked together.
121 0947 1  FORMAL PARAMETERS:
122 0948 1
123 0949 1  FMP.ra.v      Address of the user's frame.
124 0950 1
125 0951 1  IMPLICIT INPUTS:
126 0952 1
127 0953 1  The LUNs for BASIC 'channel 0'
128 0954 1
129 0955 1  IMPLICIT OUTPUTS:
130 0956 1
131 0957 1  The LUNs for BASIC 'channel 0'
132 0958 1
133 0959 1  ROUTINE VALUE:
134 0960 1  COMPLETION CODES:
135 0961 1
136 0962 1  NONE
137 0963 1
138 0964 1  SIDE EFFECTS:
139 0965 1
140 0966 1  Disables ASTs during most of its execution.
141 0967 1  OPENS two LUNs. Any errors encountered are signaled.
142 0968 1
143 0969 1  --
144 0970 1
145 0971 2  BEGIN
146 0972 2
147 0973 2  GLOBAL REGISTER
148 0974 2  CCB = K_CCB_REG : REF BLOCK [, BYTE];
149 0975 2
150 0976 2  MAP
151 0977 2  FMP : REF BLOCK [, BYTE];
152 0978 2
153 0979 2  LOCAL
154 0980 2  AST STATUS,
155 0981 2  INPUT_CCB : REF BLOCK [, BYTE],
156 0982 2  OUTPUT_CCB : REF BLOCK [, BYTE];
157 0983 2
158 0984 2  !+
159 0985 2  We are called only if one of the LUNs on channel 0 is not
160 0986 2  open, but we don't want to depend on which, so we will call
161 0987 2  BASS$CB_PUSH for each LUN, thereby using recursive I/O.
162 0988 2  First get the CCB for the input side of channel 0.
163 0989 2  --
164 0990 2  BASS$CB_PUSH (LUB$K_LUN_INPU, LUB$K_ILUN_MIN);
165 0991 2  CCB [ISB$A_USER_FP] = .FMP [SF$L_SAVE_FP];
166 0992 2  INPUT_CCB = .CCB;
167 0993 2  !+

```



```
168 0994 2 ! Now get the CCB for the output side of channel 0.
169 0995 2 !-
170 0996 2     BAS$$CB_PUSH (LUB$K_LUN BPRI, LUB$K_ILUN MIN);
171 0997 2     CCB [ISB$A_USER_FP] = .FMP [SF$L_SAVE_FP];
172 0998 2     OUTPUT_CCB = .CCB;
173 0999 2 !+
174 1000 2 ! OPEN the two LUNs. Since only this routine opens channel 0,
175 1001 2 ! and since it is not closed until image exit, both LUNs should
176 1002 2 ! be closed. If an AST causes us to re-enter this code we can
177 1003 2 ! get into serious trouble with RMS, so we must (regretfully)
178 1004 2 ! disable ASTs during the two OPENS.
179 1005 2 !-
180 1006 2     AST_STATUS = $SETAST (ENBFLG = 0);
181 1007 2
182 1008 2     IF ( NOT .INPUT_CCB [LUB$V_OPENED])
183 1009 2     THEN
184 1010 2         BEGIN
185 1011 2             CCB = .INPUT_CCB;
186 1012 2             BAS$$OPEN_DEFLT ();
187 1013 2             CCB = .OUTPUT_CCB;
188 1014 2             BAS$$OPEN_DEFLT ();
189 1015 2 !+
190 1016 2 ! Now link together the two LUNs so they can share information
191 1017 2 ! easily.
192 1018 2 !-
193 1019 2     INPUT_CCB [LUB$A_BUDDY_PTR] = .OUTPUT_CCB;
194 1020 2     OUTPUT_CCB [LUB$A_BUDDY_PTR] = .INPUT_CCB;
195 1021 2     END;
196 1022 2
197 1023 2 !+
198 1024 2 ! Now that the LUNs are set up, we can re-enable ASTs.
199 1025 2 !-
200 1026 2
201 1027 2     IF (.AST_STATUS EQL SS$_WASSET) THEN $SETAST (ENBFLG = 1);
202 1028 2
203 1029 2 !+
204 1030 2 ! Release the two CCBs, in the proper order.
205 1031 2 !-
206 1032 2     CCB = .OUTPUT_CCB;
207 1033 2     BAS$$CB_POP ();
208 1034 2     CCB = .INPUT_CCB;
209 1035 2     BAS$$CB_POP ();
210 1036 2 !+
211 1037 2 ! Our caller, who is holding the address of one of those CCBs,
212 1038 2 ! should now find that it is open.
213 1039 2 !-
214 1040 1     END;
```

! of routine BAS\$\$OPEN_ZERO

```
.TITLE BAS$$OPEN_ZERO
.IDENT \1-002\

.EXTRN BAS$$CB_PUSH, BAS$$CB_POP
.EXTRN BAS$$OPEN_DEFLT
.EXTRN SYS$SETAST

.PSECT _BAS$CODE, NOWRT, SHR, PIC, 2
```


09FC 00000				.ENTRY	BASS\$OPEN_ZERO, Save R2,R3,R4,R5,R6,R7,R8,-	
58	00000000G	00	9E 00002	MOVAB	BASS\$CB_PUSH, R8	0937
57	00000000G	00	9E 00009	MOVAB	BASS\$CB_POP, R7	
56	00000000G	00	9E 00010	MOVAB	BASS\$OPEN_DEFLT, R6	
55	00000000G	00	9E 00017	MOVAB	SYSS\$SETAST, R5	
50		08	CE 0001E	MNEGL	#8, R0	0990
52		07	CE 00021	MNEGL	#7, R2	
		68	16 00024	JSB	BASS\$CB_PUSH	
53	04	AC	DO 00026	MOVL	FMP, R3	0991
FF4C	OC	A3	DO 0002A	MOVL	12(R3), -180(CCB)	
54		5B	DO 00030	MOVL	CCB, INPUT_CCB	0992
50		08	CE 00033	MNEGL	#8, R0	0996
52		08	CE 00036	MNEGL	#8, R2	
		68	16 00039	JSB	BASS\$CB_PUSH	
FF4C	OC	A3	DO 0003B	MOVL	12(R3), -180(CCB)	0997
52		5B	DO 00041	MOVL	CCB, OUTPUT_CCB	0998
		7E	D4 00044	CLRL	-(SP)	1006
65		01	FB 00046	CALLS	#1, SYSS\$SETAST	
53		50	DO 00049	MOVL	R0, AST_STATUS	
14	FC	A4	E8 0004C	BLBS	-4(INPUT_CCB), 1\$	1008
5B		54	DO 00050	MOVL	INPUT_CCB, CCB	1011
66		00	FB 00053	CALLS	#0, BASS\$OPEN_DEFLT	1012
5B		52	DO 00056	MOVL	OUTPUT_CCB, CCB	1013
66		00	FB 00059	CALLS	#0, BASS\$OPEN_DEFLT	1014
B8	A4	52	DO 0005C	MOVL	OUTPUT_CCB, -72(INPUT_CCB)	1019
B8	A2	54	DO 00060	MOVL	INPUT_CCB, -72(OUTPUT_CCB)	1020
09		53	D1 00064	1\$: CMPL	AST_STATUS, #9	1027
		05	12 00067	BNEQ	2\$	
		01	DD 00069	PUSHL	#1	
65		01	FB 0006B	CALLS	#1, SYSS\$SETAST	
5B		52	DO 0006E	2\$: MOVL	OUTPUT_CCB, CCB	1032
		67	16 00071	JSB	BASS\$CB_POP	1033
5B		54	DO 00073	MOVL	INPUT_CCB, CCB	1034
		67	16 00076	JSB	BASS\$CB_POP	1035
		04	00078	RET		1040

; Routine Size: 121 bytes, Routine Base: _BASS\$CODE + 0000

:	215	1041	1
:	216	1042	1 END
:	217	1043	1
:	218	1044	0 ELUDOM

! of module BASS\$OPEN_ZERO

PSECT SUMMARY

Name	Bytes	Attributes
_BASS\$CODE	121	NOVEC, NOWRT, RD, EXE, SHR, LCL, REL, CON, PIC, ALIGN(2)

Library Statistics

File	----- Total	Symbols Loaded	----- Percent	Pages Mapped	Processing Time
;\$255\$DUA28:[SYSLIB]STARLET.L32;1	9776	5	0	581	00:01.2

COMMAND QUALIFIERS

; BLISS/CHECK=(FIELD,INITIAL,OPTIMIZE)/NOTRACE/LIS=LIS\$:BASOPENZE/OBJ=OBJ\$:BASOPENZE MSRC\$:BASOPENZE/UPDATE=(ENH\$:BASOPENZE
;)

; Size: 121 code + 0 data bytes
; Run Time: 00:09.0
; Elapsed Time: 00:20.8
; Lines/CPU Min: 6929
; Lexemes/CPU-Min: 42756
; Memory Used: 122 pages
; Compilation Complete

0029 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

BASOPEN
LIS

BASPOWJ
LIS

BASPOS
LIS

BASPOWJ
LIS

BASOPENDE
LIS

BASPOWGG
LIS

BASPOWH
LIS

BASPOWRJ
LIS

BASPOWII
LIS

BASPURIOB
LIS

BASPOWDO
LIS

BASOPENZE
LIS

BASPOWR
LIS

BASPOWJ
LIS

BASPOWR
LIS

BASPOWH
LIS

BASPOWR
LIS